Eur päisches Patentamt

Eur pean Patent Office

Office eur péen des brevets

J1002 U.S. PTO
10/068370
02/06/02

Bescheinigung

Certificate

Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

Patentanmeldung Nr. **Patent application No.** **Demande de brevet n°**

01103064.0

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

[illegible]

I.L.C. HATTEN-HECKMAN

DEN HAAG, DEN
THE HAGUE, 03/09/01
LA HAYE, LE

Europäisches Patentamt | European Patent Office | Office europ péen des brevets

Blatt 2 der Bescheinigung
Sheet 2 of the certificate
Page 2 de l'attestation

Anmeldung Nr.:
Application no.: 01103064.0
Demande n°:

Anmeldetag:
Date of filing: 09/02/01
Date de dépôt:

Anmelder:
Applicant(s):
Demandeur(s):
International Business Machines Corporation
Armonk, NY 10504
UNITED STATES OF AMERICA

Bezeichnung der Erfindung:
Title of the invention:
Titre de l'invention:
Controlling commands in workflow management systems

In Anspruch genommene Prioriät(en) / Priority(ies) claimed / Priorité(s) revendiquée(s)

Staat: State: Pays:	Tag: Date: Date:	Aktenzeichen: File no. Numéro de dépôt:

Internationale Patentklassifikation:
International Patent classification:
Classification internationale des brevets:
/

Am Anmeldetag benannte Vertragstaaten:
Contracting states designated at date of filing: AT/BE/CH/CY/DE/DK/ES/FI/FR/GB/GR/IE/IT/LI/LU/MC/NL/PT/SE/TR
Etats contractants désignés lors du depôt:

Bemerkungen:
Remarks:
Remarques:

D E S C R I P T I O N

Controlling Commands in Workflow Management Systems

1. Background of the Invention

1.1 Field of the Invention

The present invention relates to means and a method for improving selective command control related to the execution of instances of process models and/or activities within a Workflow-Management-System or a computer system with comparable functionality (WFMS).

1.2 Description and Disadvantages of Prior Art

A new area of technology with increasing importance is the domain of Workflow-Management-Systems (WFMS). WFMS support the modeling and execution of business processes. Business processes executed within a WFMS environment control which piece of work of a network of pieces of work will be performed by whom and which resources are exploited for this work. The individual pieces of work might be distributed across a multitude of different computer systems connected by some type of network.

The product "IBM MQSeries Workflow" (previously called IBM FlowMark) represents such a typical modern, sophisticated, and powerful workflow management system. It supports the modeling of business processes as a network of activities. This network of activities, the **process model**, is constructed as a directed, acyclic, weighted, colored graph. The nodes of the graph represent the **activities** which are performed. The edges of the graph, the **control connectors**, describe the potential sequence of execution of the activities. Definition of the process graph is via IBM MQSeries Workflow's Flow Definition Language (FDL) or via the built-in graphical editor. The runtime component of the workflow management system interprets the process graph and distributes the execution of activities to the right person at the right place, e. g. by assigning tasks in the form of

workitems to one or more worklists associated with the respective person, wherein said worklists and workitems are stored as digital data within said workflow or process management system.

Besides interacting with workitems created from an executing process instance the state of the art technology also offers the possibility to interact with a process instance by entering control commands. Examples of such commands are for instance TERMINATE and SUSPEND with the obvious meaning. Such control commands can be entered at any time during the execution of a business process provided the user who issues the command has the appropriate privileges. Thus, a privileged user can issue a command, such as TERMINATE, at any time causing consequences he didn't intend as he cannot be aware of all details which are manipulated during the execution of a certain process model. For example, some business processes cannot be terminated any more after they have carried out a particular activity or such business processes must not be terminated to not jeopardize the consistency of the data such a process is manipulating (for instance because data may irreversibly be modified without any rollback operation possibility). It is evident that a user or even a trained administrator cannot foresee all consequences of sending a certain control command to a process instance.

As a result the capability to issue any command at any time is not always desirable. Moreover the knowledge available only to the development team of a certain business process which control commands can be executed at which processing stages of the corresponding process model without creating any harm to the overall business result somehow has to be "Enabled" to allow a business process to protect itself from the execution of non permissible control commands.

The weakness of the state of the art approach with respect to this problem area becomes even more distinct if one thinks of typical Internet scenarios commonly summarized by terms like C2B

(Consumer-to-Business) or B2B (Business-to-Consumer) business processes. Due to business reasons also within such scenarios certain privileges must be assigned to the consumer or business that initiated a business process at a company. Without further protection a computer illiterate clerk who drives such a business process could jeopardize the consistency of the overall system by issuing such control commands.

1.3 Objective of the Invention

The invention is based on the objective to reduce the risk against control commands issued against a process instance executed under the control of Workflow Management System (WFMS) which would "harm" the underlying business process or jeopardize the consistency of the corresponding business process data.

2. Summary and Advantages of the Invention

The objectives of the invention are solved by the independent claims. Further advantageous arrangements and embodiments of the invention are set forth in the respective subclaims.

The present invention relates to a method and to a system for providing selective command control within a Workflow-Management-System or a computer system with comparable functionality (WFMS). It is assumed that the WFMS comprises a process model of a business process and the process model comprise one or more activities being the nodes of an arbitrary graph, and directed edges of said graph defining a potential control flow within said process model. Upon receiving an issued command directed to a process instance of said process model the method and corresponding system in a first step is determining if a current activity instance, currently having control within the flow of control through the process instance, is comprised by a command sphere. The command sphere comprises a sub-graph of said arbitrary graph and defines one or a multitude of permissible commands allowed to be executed or not allowed to be executed if control resides within said commands sphere. In a second step the method and corresponding system is executing said issued command only, if it is a permissible command.

The suggested approach significantly reduces the risk that a user can jeopardize the consistency of the overall business process by issuing certain control commands. The proposed technology allows that for instance the development team of a particular business process which has the most thorough understanding of the processing details can define a command sphere providing a self-protecting mechanism for the underlying business process against issued control commands by any user. This self-protecting mechanism is operating selectively as the permissible commands are dependent on the particular activity which currently has control within the flow of control through the process model.
Based on the teaching of command spheres a technology is provided which allows to model dependencies within a process model which up to now have been out of the realm of Workflow Management Systems.

3. Brief Description of the Drawings

Figure 1 shows an example of a process model represented by a process graph.

Figure 2 illustrates on an exemplary level that in addition to the states a certain process instance may occupy while the flow of control is moving through the process graph a process instance can occupy various further states when it is carried out by the workflow management system.

Figure 3 reflects an example of a process model representing a book order process comprising a single command sphere to enable a user to cancel a certain book order only before the book has been shipped (but not in other stages of the execution of the process model).

Figure 4 visualizes the details of the specification of this command sphere for the process model of Fig. 3 using the Flow Definition Language of MQSeries Workflow.

4. Description of the Preferred Embodiment

In the drawings and specification there has been set forth a preferred embodiment of the invention and, although specific terms are used, the description thus given uses terminology in a generic and descriptive sense only and not for purposes of limitation. It will, however, be evident that various modifications and changes may be made thereto without departing from the broader spirit and scope of the invention as set forth in the appended claims.

The present invention can be realized in hardware, software, or a combination of hardware and software. Any kind of computer system - or other apparatus adapted for carrying out the methods described herein - is suited. A typical combination of hardware and software could be a general purpose computer system with a computer program that, when being loaded and executed, controls the computer system such that it carries out the methods described herein. The present invention can also be embedded in a computer program product, which comprises all the features enabling the implementation of the methods described herein, and which - when being loaded in a computer system - is able to carry out these methods.

Computer program means or computer program in the present context mean any expression, in any language, code or notation, of a set of instructions intended to cause a system having an information processing capability to perform a particular function either directly or after either or both of the following a) conversion to another language, code or notation; b) reproduction in a different material form.

The current invention is illustrated based on IBM's "MQSeries Workflow" workflow management system. Of course any other WFMS could be used instead. Furthermore the current teaching applies also to any other type of system which offers WFMS functionalities not as a separate WFMS but within some other type of system.

Though the control commands of the following examples targeting at a certain running process model are processed by the WFMS engine this should not be understood as a limitation. The current invention can be applied in other scenarios wherein the processing entity of the control commands is not the WFMS engine itself.

4.1 Introduction

The following is a short outline on the basic concepts of a workflow management system based on IBM's "MQSeries Workflow" WFMS:

From an enterprise point of view the management of business processes is becoming increasingly important: **business processes** or **process** for short control which piece of work will be performed by whom and which resources are exploited for this work, i.e. a business process describes how an enterprise will achieve its business goals. A WFMS may support both, the modeling of business processes and their execution.

Modeling of a business process as a syntactical unit in a way that is directly supported by a software system is extremely desirable. Moreover, the software system can also work as an interpreter basically getting as input such a model: The model, called a **process model** or **workflow model**, can then be instantiated and the individual sequence of work steps depending on the context of the instantiation of the model can be determined. Such a model of a business process can be perceived as a template for a class of similar processes performed within an enterprise; it is a schema describing all possible execution variants of a particular kind of business process. An instance of such a model and its interpretation represents an individual process, i.e. a concrete, context dependent execution of a variant prescribed by the model. A WFMSs facilitates the management of business processes. It provides a means to describe models of business processes (buildtime) and it drives business processes based on an associated model (runtime). The meta model of IBM's WFMS MQSeries Workflow, i.e. the syntactical elements provided for describing business process models, and the meaning and interpretation of these syntactical elements, is described next.

A process model is a complete representation of a process, comprising a process diagram and the settings that define the logic behind the components of the diagram. Important components of a MQSeries Workflow process model are:

- Processes
- Activities
- Blocks
- Control Flows
- Connectors
- Data Containers
- Data Structures
- Conditions
- Programs
- Staff

Not all of these elements will be described below.

Activities are the fundamental elements of the meta model. An activity represents a business action that is from a certain perspective a semantic entity of its own.

A MQSeries Workflow process model consists of the following types of activities:
Program activity: Has a program assigned to perform it. The program is invoked when the activity is started. In a fully automated workflow, the program performs the activity without human intervention. Otherwise, the user must start the activity by selecting it from a runtime work list. Output from the

program can be used in the exit condition for the program activity and for the transition conditions to other activities. **Process activity**: Has a (sub-)process assigned to perform it. The process is invoked when the activity is started. A process activity represents a way to reuse a set of activities that are common to different processes. Output from the process, can be used in the exit condition for the process activity and for the transition conditions to other activities.

The flow of control, i.e. the **control flow** through a running process determines the sequence in which activities are executed. The MQSeries Workflow workflow manager navigates a path through the process that is determined by the evaluation to TRUE of start conditions, exit conditions, and transition conditions.

Connectors link activities in a process model. Using connectors, one defines the sequence of activities and the transmission of data between activities. Since activities might not be executed arbitrarily they are bound together via **control connectors**. A control connector might be perceived as a directed edge between two activities; the activity at the connector's end point cannot start before the activity at the start point of the connector has finished (successfully). Control connectors model thus the potential flow of control within a business process model. Default connectors specify where control should flow when the transition condition of no other control connector leaving an activity evaluates to TRUE. Default connectors enable the workflow model to cope with exceptional events. Data connectors specify the flow of data in a workflow model. A data connector originates from an activity or a block, and has an activity or a block as its target. One can specify that output data is to go to one target or to multiple targets. A target can have more than one incoming data connector.

Process definition includes modeling of activities, control connectors between the activities, input/output container, and data connectors. A process is represented as a directed acyclic graph with the activities as nodes and the control/data connectors as the edges of the graph. The graph is manipulated via a built-in graphic editor. The data containers are specified as named data structures. These data structures themselves are specified via the DataStructureDefinition facility. Program activities are implemented through programs. The programs are registered via the Program Definition facility. Blocks contain the same constructs as processes, such as activities, control connectors etc. They are however not named and have their own exit condition. If the exit condition is not met, the block is started again. The block thus implements a Do Until construct. Process activities are implemented as processes. These subprocesses are defined separately as regular, named processes with all its usual properties. Process activities offer great flexibility for process definition. It not only allows to construct a process through permanent refinement of activities into program and process activities (top-down), but also to build a process out of a set of existing processes (bottom-up).

All programs which implement program activities are defined via the Program Registration Facility. Registered for each program is the name of the program, its location, and the invocation string. The invocation string consists of the program name and the command string passed to the program.

As an example of such a process model Fig. 1 shows schematically the structure of such a process graph. Activities (A1 up to A5) are represented as named circles; the name typically describes the purpose of the activity. Activities come in various flavors to address the different tasks that may need to be performed. They may have different activity implementations to meet these diverse needs. **Program activities** are performed by an assigned program, **process activities** like for instance 100 are performed by another process 101, and **blocks** like for instance 102 implement a macro 103 with a built-in do-until loop.
Control connectors p12, p13, p24, p35, p45 are represented as

arrows; the head of the arrow describes the direction in which the flow of control is moving through the process. The activity where the control connector starts is called the **source activity**; where it ends is called the **target activity**. When more than one control connector leaves an activity, this indicates potentially parallel work.

4.2 Process States

In addition to the states a certain process instance may occupy while the flow of control is moving through the process graph a process instance can occupy various further states when it is carried out by the workflow management system. Fig. 2 illustrates those states exemplary. It should be noted that this for illustration purpose only; workflow management systems typically differentiate between many more states.

The first step a particular business process goes through is that it is created by taking the appropriate process template, possibly populating it with supplied context data, and assigning it a unique process instance identification. This step is usually carried out as the result of invoking the workflow management system's CREATE function. As a result of function completion, the business process is put into the state **created** 201 creating a process instance from a process model (the template).

When the business process is being carried out, that means the workflow management system navigates through the process graph and executes the individual activities, the business process is in the state **running** 202. The business process is typically put into this state by a client issuing a START control command; other possibilities are that the business process is automatically started by the workflow management system at a time specified when the business process is created, or a combination of a CREATE and START control command.

When all activities of the business process have been carried out, the process goes into the state **finished** 203. No further activities are carried out with the business process; however all information about the business process is still available and can for example be queried. Some workflow management system still allow operations on a finished business process, such as restarting the business process at the beginning or even in the middle of the business process.

No further actions can be carried out if the business process is in the state **deleted** 204. Whether all the business process's information is removed immediately from the workflow management system's store depends on the actual implementation; some workflow management systems do, some require the invocation of a DELETE function by a corresponding control command.

The state **suspended** 205 is entered as the result of entering the SUSPEND function by issuing a corresponding control command. In this state, the workflow management system no longer navigates the business process until requested by a user via the RESUME control command.

A business process enters the state **terminated** 206 as the result of the TERMINATE control command, which causes the workflow management system to stop processing the business process.

4.3 Command Spheres

As outlined already within above description of prior art the capability to issue any control command directed to a certain process instance at any time is not always desirable. For instance process instance implementing a business process in an Internet scenario commonly summarized by terms like C2B (Consumer-to-Business) or B2B (Business-to-Consumer) business processes are driven by computer illiterate clerks who easily could jeopardize the consistency of the overall business process by issuing such control commands.

In a first observation it is therefore suggested to "enabled" a business process to protect itself from the execution of non permissible control commands. As in a second observation the knowledge, which control commands can be executed at which processing stages of the corresponding process model without creating any harm to the overall business result, typically is available only to the development team of a certain business process the suggested technology allows to specify for each processing stage of a certain process model the precise set of control commands which permissible may be processed in this stage. As ideal place for storing these specifications the process model itself is suggested as the process model is set up already by the development team.

The technology to specify, which control commands can be executed at which processing stages of a certain process model without creating any harm to the overall business result, is the concept of **command spheres**. A command sphere identifies a set of activities within a process model and defines for this set of activities which control commands can or cannot be issued by a user if any of these activities having control within the flow of control through the process instance. In case the command sphere is defined in terms of non-permissible control commands it is further suggested that for each of the non-valid commands an action may be defined that should be carried out in case the invalid command is entered by the user; thus, this teaching allows to define an "substitute" action to be performed in case a non permissible control command has been issued. In general a commands sphere may comprise any sub graph of the process model. Fig. 3 reflects an example of a process model representing a book order process. To enable a user to cancel a certain book order any time before the book has been shipped (but not in other stages of the execution of the process model) command sphere 301 has been defined. The details of the specification of this command sphere for the process model of Fig. 3 are illustrated using the Flow Definition Language of MQSeries Workflow within Fig. 4.

The command sphere 301 comprises the activities "Ship book" 302 and the activity "Debit credit card" 303 with the intention to specify that once the flow of control resides in any of these activities of the book order business process the canceling of the order (expressed by issuing the TERMINATE command) is no longer permissible.

Inspecting the Float Definition Language it becomes apparent that a new section COMMAND_SPHERE 401 has been added that allows to identify a sub graph in the process model, representing the specification of the command sphere "CannotCancelOrderAnymore". The keyword NON_VALID_COMMANDS 402 provides for the specification of parameters which identify those commands that are not valid within the command sphere (definition of the non-permissible commands). The ACTION keyword 403 provides, for each of the non-valid commands, the specification of an action that should be carried out if the non-permissible command is issued by a user (this represents the substitute action mentioned above). The action could be anything that can be carried out by the workflow management system, such as a program, a process, or even a further command; in the current example the substitute action 404 consists in sending an e-mail. The identification of those activities which belong to the commands sphere takes place within the specification sections of the individual activities. Referring to the current example of Fig. 4 the specification section 405 of the activity "Ship book" comprises the RELATED_COMMAND_SPHERE statement 406, which identifies this activity as belonging to the command sphere CannotCancelOrderAnymore 401; similar the specification section 407 of the activity "Debit credit card" comprises the RELATED_COMMAND_SPHERE statement 408, which identifies this activity as belonging to the command sphere CannotCancelOrderAnymore 401.

It can be noted that the complete process model is conceptually a command sphere wherein all commands of the workflow management system are supported.

As further embodiment of the current invention further methods are suggested to specify that certain commands are not supported for certain pieces (sub graphs) of the business process.

a. One method is to allow for the specification of the valid or non-valid commands for each individual activity (that is, within the specification sections of the individual activities). This approach is equivalent of having a command sphere that includes only the appropriate activity.

b. Another method is to attach the valid or non-valid commands to an activity and have that specification valid until overwritten by another specification or by the end of the process. This can be easily transformed into appropriate command sphere specifications as discussed above.

Thus, all these variants do not deviate from the current teaching of command spheres as a conceptual approach. These variants are simply based on different approaches relating to the specification techniques of command spheres.

With respect to further embodiments of the current invention it is outlined next that there are almost no limitations to the structure of command spheres: command spheres may include other command spheres or may even overlap with other command spheres. If a first command sphere is completely comprised by a second command sphere, this can be interpreted that the permissible commands defined in the first command sphere will override the second permissible commands of the second command sphere.
If a first command sphere is overlapping with a third command sphere and a control command issued while control resides in a process instance which is comprised by the first command sphere as well as by the third command sphere, this can be interpreted that the issued control command is executed only if it is a permissible command of said first command sphere as well as of said third command sphere.

C L A I M S

1. A computerized method of providing selective command control within a Workflow-Management-System or a computer system with comparable functionality (WFMS),

said WFMS comprising a process-model of a business process, said process-model comprising one or more activities being the nodes of an arbitrary graph, and directed edges of said graph defining a potential control-flow within said process-model, and

said method upon receiving an issued command directed to a process-instance of said process-model,

in a first step determining if a current activity instance, currently having control within the flow of control through the process-instance, is comprised by a command-sphere,

said command-sphere comprising a sub-graph of said arbitrary graph, and

said command-sphere defining one or a multitude of permissible commands allowed to be executed if control resides within said command-sphere; and

in a second-step executing said issued command, if it is a permissible command.

2. A computerized method of providing selective command control within a WFMS according to claim 1,

wherein said command-sphere defining at least one substitute-action to be performed in case of a non-permissible command, and

said method in a third step, if said issued command is a non-permissible command, performing said substitute-action instead of said issued command.

3. A computerized method of providing selective command control within a WFMS according to claim 2,

wherein said command-sphere is completely comprised by a second command-sphere, and

wherein in said first step said permissible commands defined in said command-sphere overriding second permissible commands of said second command-sphere.

4. A computerized method of providing selective command control within a WFMS according to claim 2,

wherein said command-sphere is overlapping with a third command-sphere, and

wherein in said first step said current activity-instance is comprised by said command-sphere as well as said third command-sphere, and

wherein in said second step said issued command is executed only if it is a permissible command of said command-sphere as well as of said third command-sphere.

5. A computerized method of providing selective command control within a WFMS according to anyone of the preceding claims,

wherein said permissible commands are defined

either by explicitly specifying allowed commands,

or by the complement of the explicitly specified not-allowed commands.

6. A computerized method of providing selective command control within a WFMS according to anyone of the preceding

claims,

EPO - Munich
63
09. Feb. 2001

wherein said method is executed by the WFMS itself.

7. A computerized method of providing selective command control within a WFMS according to anyone of the preceding claims,

wherein said method command-spheres are specified within said process-model of said business process.

8. A system comprising means adapted for carrying out the steps of the method according to anyone of the preceding claims 1 to 7.

9. A data processing program for execution in a data processing system comprising software code portions for performing a method according to anyone of the preceding claims 1 to 7 when said program is run on said computer.

10. A computer program product stored on a computer usable medium, comprising computer readable program means for causing a computer to perform a method according to anyone of the preceding claims 1 to 7 when said program is run on said computer.

EPO - Munich
63
09. Feb. 2001

A B S T R A C T

The present invention relates to a method and to a system for providing selective command control within a Workflow-Management-System or a computer system with comparable functionality (WFMS). It is assumed that the WFMS comprises a process model of a business process and the process model comprise one or more activities being the nodes of an arbitrary graph and directed edges of said graph defining a potential control flow within said process model. Upon receiving an issued command directed to a process instance of said process model the method and corresponding system in a first step is determining if a current activity instance, currently having control within the flow of control through the process instance, is comprised by a command sphere. The command sphere comprises a sub-graph of said arbitrary graph and defines one or a multitude of permissible commands allowed to be executed or not allowed to be executed if control resides within said commands sphere. In a second step the method and corresponding system is executing said issued command only, if it is a permissible command. (Fig. 3)

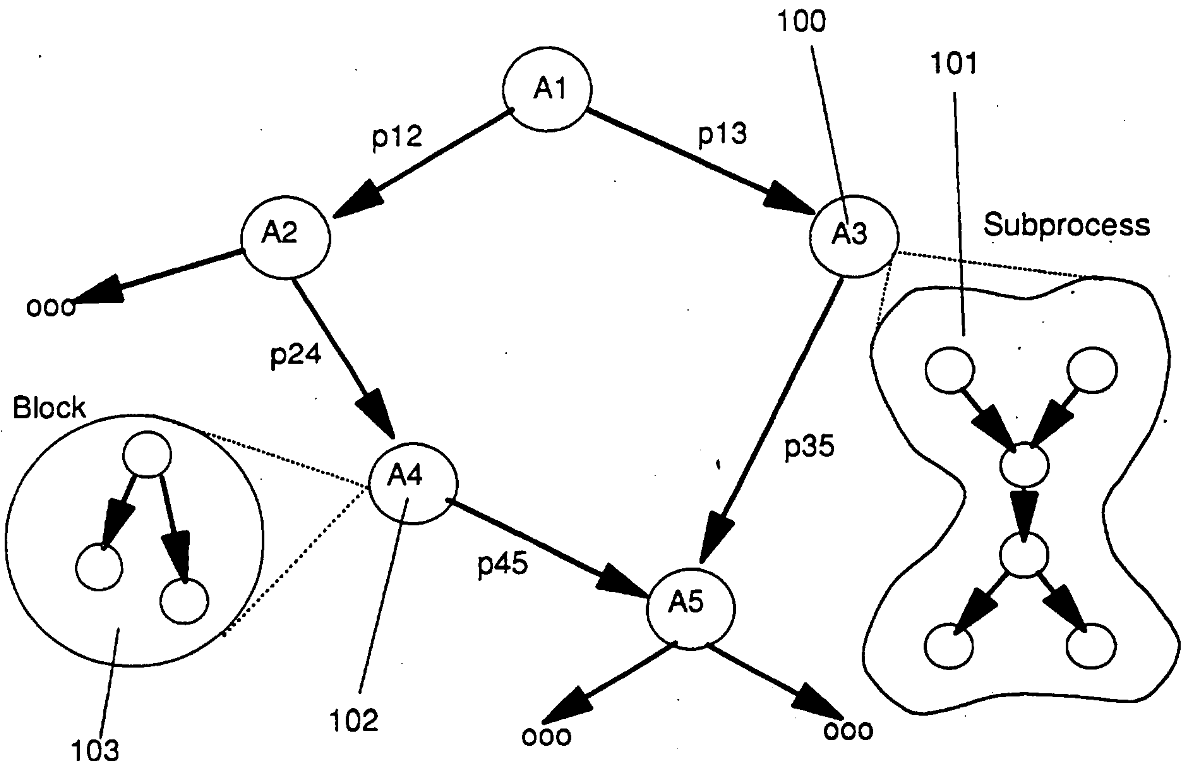

A1
A2
A3
A4
A5
p12
p13
p24
p35
p45
ooo
ooo
ooo
100
101
102
103
Block
Subprocess

FIG. 1

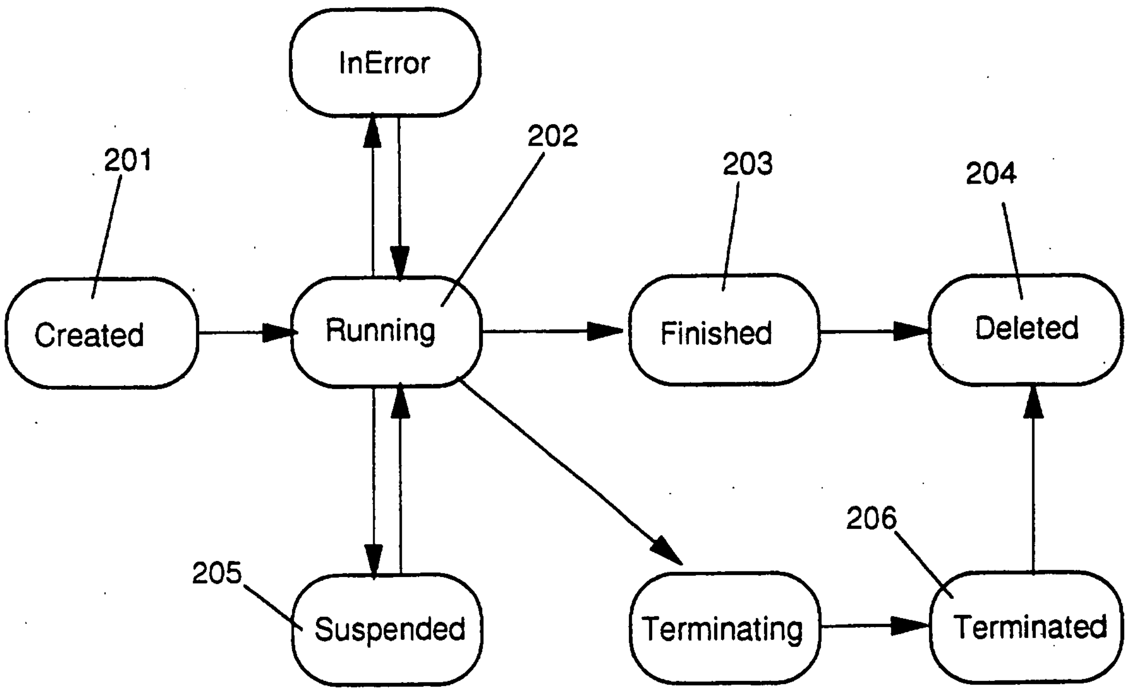

InError
201
202
203
204
Created
Running
Finished
Deleted
206
205
Suspended
Terminating
Terminated

FIG. 2

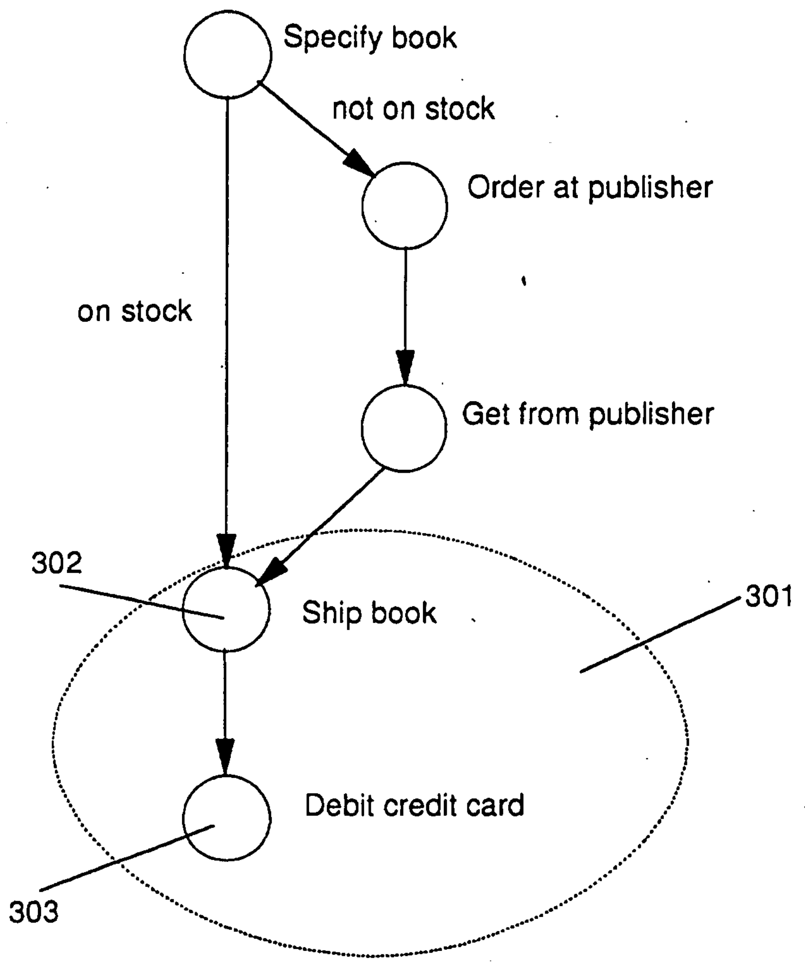
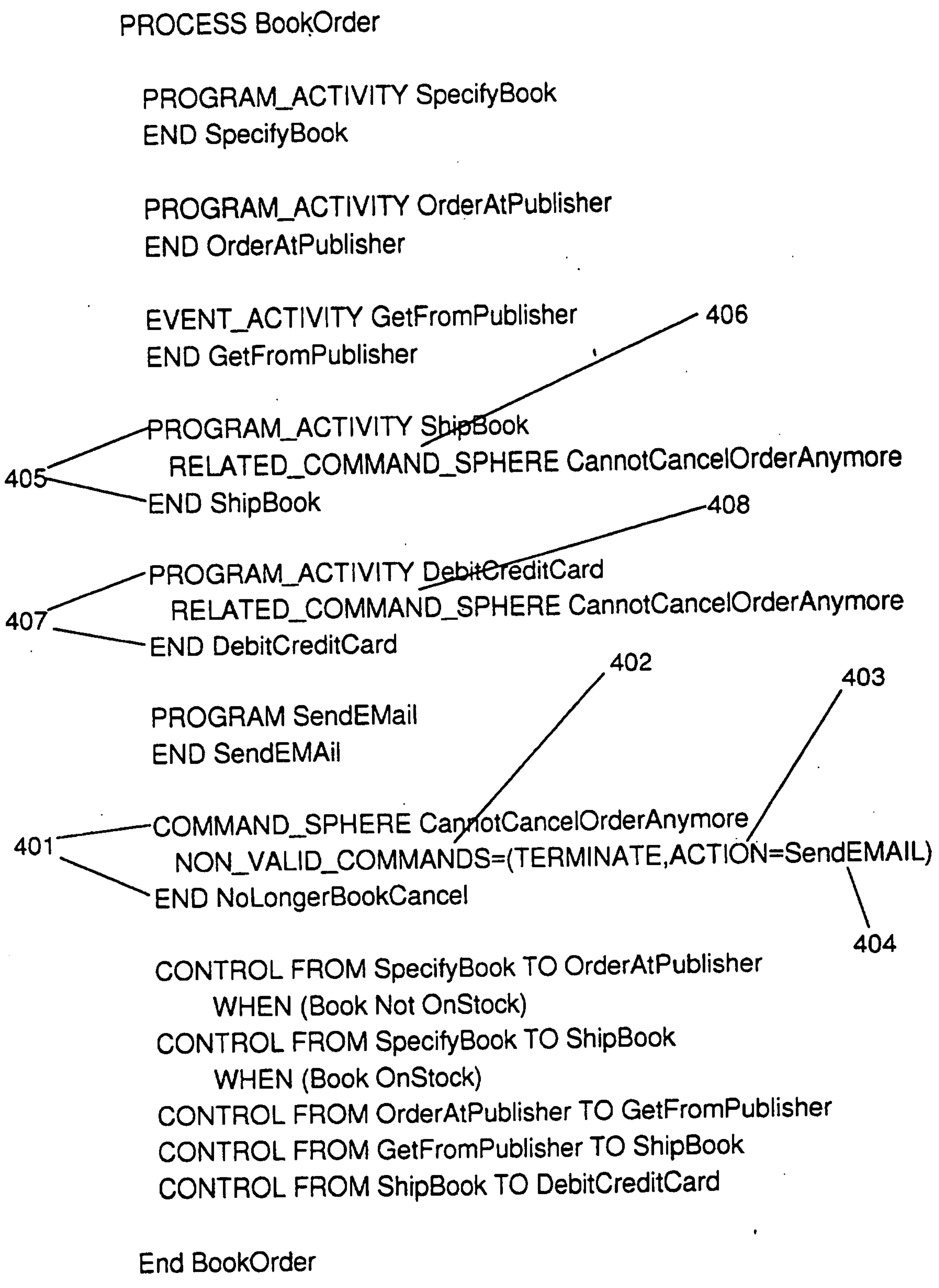

Specify book
not on stock
Order at publisher
on stock
Get from publisher
302
Ship book
301
Debit credit card
303

FIG. 3


406
405
408
407
402
403
401
404

```
PROCESS BookOrder

  PROGRAM_ACTIVITY SpecifyBook
  END SpecifyBook

  PROGRAM_ACTIVITY OrderAtPublisher
  END OrderAtPublisher

  EVENT_ACTIVITY GetFromPublisher
  END GetFromPublisher

  PROGRAM_ACTIVITY ShipBook
    RELATED_COMMAND_SPHERE CannotCancelOrderAnymore
  END ShipBook

  PROGRAM_ACTIVITY DebitCreditCard
    RELATED_COMMAND_SPHERE CannotCancelOrderAnymore
  END DebitCreditCard

  PROGRAM SendEMail
  END SendEMAil

  COMMAND_SPHERE CannotCancelOrderAnymore
    NON_VALID_COMMANDS=(TERMINATE,ACTION=SendEMAIL)
  END NoLongerBookCancel

  CONTROL FROM SpecifyBook TO OrderAtPublisher
        WHEN (Book Not OnStock)
  CONTROL FROM SpecifyBook TO ShipBook
        WHEN (Book OnStock)
  CONTROL FROM OrderAtPublisher TO GetFromPublisher
  CONTROL FROM GetFromPublisher TO ShipBook
  CONTROL FROM ShipBook TO DebitCreditCard

End BookOrder
```

FIG. 4